[redacted]

510.84
Il6r
no.649 UIUCDCS-R-74-649

9

[redacted]

MINIMIZATION OF LOGIC NETWORKS UNDER A GENERALIZED COST FUNCTION*

by

Saburo Muroga
Hung Chi Lai

April, 1974

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

UIUCDCS-R-74-649

MINIMIZATION OF LOGIC NETWORKS UNDER A GENERALIZED COST FUNCTION*

by

Saburo Muroga
Hung Chi Lai

*This work was supported in part by the National Science Foundation under Grant No. NSF-GJ-40221.

I. INTRODUCTION

Recent progress in integrated circuit (IC) technology has brought out many new aspects to be considered in the logical design of digital networks. One of them is the desirability of implementing a compact network for a given switching function. Conventionally, the cost of a logical network with discrete components has been dominated by the number of gates in that network and consequently the term "a minimal network" has usually meant a network with a minimal number of gates and then, as a secondary objective, with a minimum number of connections, among all possible realizations for a given function.[11]~[15] In other words, a minimal network is the one which minimizes the cost function

$$h(G,C) = LG + C, \tag{1.1}$$

where G and C are the numbers of gates and connections in a network, respectively, and L is a sufficiently large positive number.

In the case of networks in MSI (Medium Scale Integration) or LSI (Large Scale Integration), however, the cost of a network implemented in IC is determined by many factors, such as the chip area occupied by the network, the number of masks, production yield, production volume, etc. If all the factors other than the chip area are fixed, the cost of a network implemented in IC for a given function f (here f means a switching function to be realized which, if necessary, has all desired facilities such as diagnosability) usually increases more than linearly as the chip area increases (since a larger chip area usually lowers the yield). Furthermore the chip area is covered not only by gates but also by connections. Depending on the type of electronic implementation, either the gate area or the connection area is predominant or the two are comparable in terms of chip area.

Suppose we want to minimize the gate and connection areas, seperately. Then if gates occupy a larger chip area than connections, as in the case of bipolar LSI, the minimization of the gate area as the first objective and then the minimization of the connection area as the second objective would make the chip area compact. However, if the connections occupy a larger area as in the case of MOS LSI, the minimization of the gate area as the first objective may not be appropriate, but the minimization of the connection area as the first objective would be more appropriate.

Considering these relationships, a reasonable concept of a generalized cost function to be minimized in logical design is introduced in Section II. Then the minimization problem of this cost function can be solved; by deriving networks which minimize the number of gates, first, and the number of connections, second; networks which minimize the number of connections, first, and the number of gates, second; and then certain networks which are associated to the above networks.

As a computational example, we find these three types of networks for all switching functions of three or less variables in Section III and for some functions of 4 variables in Section IV, assuming that these networks consist of NOR gates only and using the integer programming approach. The computational results in Sections III and IV indicate that these three types of networks are identical for most of the functions tested. (This would be an interesting result even from the viewpoint of classical switching theory.) This means that for most of the functions networks which minimize the cost function (1.1) minimize the generalized cost function also and consequently would yield reasonably compact networks.

II. GENERALIZED COST FUNCTION AND ITS IMPLICATIONS

Suppose a network for a given switching function f is to be implemented in integrated circuits (IC). (If the network is required to have special facilities such as diagnosability, f is the output of the network provided with such facilities. If maintenability requires extra outputs or a special network configuration, a network for f is to be implemented along with such outputs or under such a restriction of configuration, respectively. Multiple output networks can be treated in the same manner as the following.) Though the cost of networks implemented in IC is determined by many factors, let us consider in this paper only its relation to the chip area of the network, assuming that other factors such as product volume, the number of masks, and production yields are fixed(because these factors are pertinent to the production rather than to the logical design).

Then the cost of a network increases more than linearly as the chip area covered by that network increases. So it is economically desirable to minimize the chip area. However, usually it is not obvious how much chip area any given network would occupy, unless the layout of the network on a chip is actually made. Since the layout has many complex topological design restrictions such as those on cross-unders, field inversion, and heat dissipation[20], the actual layout is very tedious and time-consuming. Thus it would be desirable to establish a guide line in deriving a logical network which will yield the most compact network, without trying layouts of many other logical networks. In this section, we introduce a generalized cost function to be minimized in logical design, which we hope to be such a guide line. However, since the generalized cost function is introduced with simplification assumptions of very complicated layout restrictions (though the assumptions appear reasonable), the minimization of the generalized cost function may not always lead to the most compact networks but would lead to reasonably compact networks. Of course, to see whether it really does, we must wait for the accumulation of empirical evidence by actual layout.

Chip Area Versus the Numbers of Gates and Connections

The chip area covered by the connections in a network cannot be ignored since it is sometimes comparable to or greater than the area covered by gates, as the following examples show. In the case of MOS networks, MOSFETs occupy very small areas and connections cover a much greater area on a chip. In the case of TTL networks or ECL networks, each gate occupies a large area because each gate contains many bipolar transistors and resistors and each bipolar transistor requires an area for isolation. Thus gates occupy more area than connections.

Consequently a cost function to be introduced should be a monotonically increasing function of the connection area and the gate area on a chip. (Note that the connection and gate areas usually share a common area since connections may run over gates.) However both the connection and gate areas cannot be determined unless the layout is actually made, contradicting our objective of introducing the cost function. Thus let us consider the number of gates and the number of connections, instead of the gate area and the connection area, respectively. In other words, we want to introduce a cost function which is monotonically increasing function of the numbers of gates and connections.

If we assume that gates in a network occupy roughly equal chip areas, the replacement of the gate area by the number of gates is justifiable, since the gate area is proportional to the number of gates.

The connection area is linearly proportional to the total length of connections, so the use of the total length is desirable. But the total length also cannot be determined unless the layout is actually made. Thus let us use the total number of connections. This would be justifiable, because the total length tends to increase as the number of connections increases (i.e., as the number of connections (and/or the number of gates) increases, the network requires more area. Thus each connection tends to be longer).

It should be noticed that the reduction in the number of gates or connections yields also other things, which indirectly reduce chip area or cost. In the case of TTL networks, each connection goes to a corresponding emitter in some gate, and consequently the number of these emitters is reduced, further reducing chip area. In the case of ECL networks, each connection requires an input transistor in a gate to which the connection goes and consequently the number of these input transistors is reduced, further reducing chip area. Also in each case of ECL, TTL, or MOS networks, the reduction in the number of gates reduces the power consumption of the entire network, power supply connections, and ground connections, further reducing chip area. Also the reduction in the number of gates, connections, or both improves the production yield, further reducing chip cost.

Integrated injection logic [18],[19],[21] (or merged transistor logic) which yields compact networks with bipolar transistors is an interesting case in the following sense. As can be easily seen, an integrated injection logic network with a minimum number of bipolar transistors (excluding injectors) can be derived from a network with NOR gates only which has a minimum number of gates. When each transistor has few collectors, each transistor occupies approximately equal area. Thus NOR networks with a minimum number of gates (obtained in Sections III and IV) would yield compact integrated injection logic networks.

Generalized Cost Function

Since smaller numbers of gates and connections usually yield a more compact network as discussed above, let us introduce a cost function $h(G,C)$ which is a monotonically increasing function of G and C, where G and C are the numbers of gates and connections, respectively. Then we get the following properties.

Theorem 1: If two networks for a given function f which minimize a given monotonically increasing cost function $h(G,C)$ have different pairs, (G_1, C_1) and (G_2, C_2), then there holds either

$$G_1 > G_2 , C_1 < C_2$$

or

$$G_1 < G_2 , C_1 > C_2 .$$

Proof: Suppose $G_1 = G_2$. Then $C_1 \neq C_2$ since (G_1, C_1) and (G_2, C_2) are different. If $C_1 > C_2$, $h(G_1, C_1) > h(G_2, C_2)$ must hold since h is a monotonically increasing function. This contradicts that the network with (G_1, C_1) minimizes h. If $C_1 < C_2$, we will have a contradiction similarly. Thus $G_1 \neq G_2$.

Similarly if $C_1 = C_2$, we will have a contradiction. Thus we must similtaneously have $G_1 \neq G_2$ and $C_1 \neq C_2$.

If $G_1 < G_2$ and $C_1 < C_2$, $h(G_1, C_1) < h(G_2, C_2)$ must hold since h is a monotonically increasing function. This contradicts the assumption that the network with (G_2, C_2) minimizes h. Similarly $G_1 > G_2$ and $C_1 > C_2$ lead to a contradiction. Thus theorem statement holds. Q.E.D.

Therefore the pairs, (G_i, C_i)'s of all the networks which minimize an $h(G,C)$ can be uniquely ordered according to G_i or C_i. In other words, if G_i's are ordered in the ascending order, C_i's are ordered in the descending order, and vice versa. Here the two terminal pairs in this ordering are the most important. Let (G_g, C_g) be the pair such that $G_g \leq G_i$ and $C_g \geq C_i$ for every i. Let (G_c, C_c) be the pair such that $G_c \geq G_i$ and $C_c \leq C_i$ for every i.

<u>Corollary 1</u>: The number of pairs, (G_i, C_i)'s of all the networks which minimize a given monotonically increasing cost function $h(G,C)$ is at most $\min(G_c - G_g + 1, C_g - C_c + 1)$.

<u>Corollary 2</u>: If $(G_g, C_g) = (G_c, C_c)$, then all the networks for f which minimize a given monotonically increasing cost function $h(G,C)$ have the same pair, (G_g, C_g).

Because of Corollary 1, the number of pairs of all the networks for f which minimize an h is finite. Since the number of networks with a particular pair is finite, the total number of networks for f which minimize h is finite. Thus it is possible to list all the networks for f which minimize a generalized cost function $h(G,C)$.

Let us find a relationship between an arbitrary monotonically increasing cost function and a cost function which is a linear function of G and C, i.e.,

$$h^{o}(G,C) = AG + BC$$

where A and B are both positive numbers. This $h^{o}(G,C)$ is a special case of $h(G,C)$.

Definition 1: For a given function f, let us find a network which minimizes

$$AG + BC \qquad \text{where } A \gg B.$$

Henceforth the minimization of AG + BC under the relation $A \gg B$ will be denoted as GCM (i.e., the number of gates is minimized first and then the number of connections is minimized.). Then a solution pair is denoted with (G_g^o, C_g^o). Next let us find a network which minimizes

$$AG + BC \qquad \text{where } A \ll B.$$

Henceforth the minimization of AG + BC under the relation $A \ll B$ will be denoted as CGM (i.e., the number of connections is minimized first and then the number of gates is minimized). Then a solution pair is denoted with (G_c^o, C_c^o).

When pairs, (G_g^o, C_g^o) and (G_c^o, C_c^o), are different, $G_g^o \neq G_c^o$ and $C_g^o \neq C_c^o$ hold simultaneously, as can be easily seen.

Definition 2: Suppose (G_g^o, C_g^o) and (G_c^o, C_c^o) are different. Let S denote the set consisting of the pairs, (G_g^o, C_g^o) and (G_c^o, C_c^o). Then let us find a network of $G_g^o + 1$ gates which minimizes the number of connections. If a network is found and the minimized number of connections C_1^o is smaller than C_g^o, add the new pair $(G_g^o + 1, C_1^o)$ into S. Otherwise discard the pair. Then let us find a network of $G_g^o + 2$ gates which minimizes the number of connections. If a network is found and the minimized number of connections C_2^o is smaller than the smallest number of connections in any pair in S other than (G_c^o, C_c^o), add the new pair $(G_g^o + 2, C_2^o)$ into S. Otherwise discard the pair. Continue this procedure until a network of G_c^o gates is considered. The pairs in S are called the minimal pairs for f, or the minimal (G,C) pairs for f. Let us rewrite these pairs in S as (G_g^o, C_g^o), (G_1^o, C_1^o), (G_2^o, C_2^o), ..., (G_i^o, C_i^o), ..., (G_c^o, C_c^o) for later use (i.e., i=g, 1, 2, ..., c).

Notice that for any two minimal pairs in set S, (G_{i1}^o, C_{i1}^o) and (G_{i2}^o, C_{i2}^o), either $G_{i1}^o > G_{i2}^o$, $C_{i1}^o < C_{i2}^o$ or $G_{i1}^o < G_{i2}^o$, $C_{i1}^o > C_{i2}^o$ holds. Thus if G_i^o's are ordered in the ascending order, the corresponding C_i^o's are ordered in the descending order. Also notice that more than one network may have each pair (G_i^o, C_i^o) in S.

Lemma 1: When the pairs, (G_g^o, C_g^o) and (G_c^o, C_c^o) are different, let S' denote the set consisting of the pairs (G_g^o, C_g^o) and (G_c^o, C_c^o). Then let us find a network of $C_c^o + 1$ connections which minimizes the number of gates. If a network is found and the minimized number of gates G_t^o is smaller than G_c^o, add the new pair $(G_t^o, C_c^o + 1)$ into S'. Otherwise discard the pair. Then let us find a network of $C_c^o + 2$ connections which minimizes the number of gates. If a network is found and the minimized number of gates G_{t-1}^o is smaller than the the smallest number of gates in any pair in S other than (G_g^o, C_g^o), add the new pair $(G_{t-1}^o, C_c^o + 2)$ into S'. Otherwise discard the pair. Continue this procedure until a network of C_g^o connections is considered. Then S' is identical to S in Definition 2.

Proof: Let us consider an arbitrary pair (G_j^o, C_j^o) in S' which is formed according to Lemma 1 statement (i.e., a network with C_j^o connections minimizes the number of gates to G_j^o).

Assume that this C_j^o is not contained in any pair in S. The C_j^o is not contained in S either

(i) because during the formation of S another pair (G_j^o, C_{j1}^o) where $C_{j1}^o < C_j^o$ was found to exist (for this proof we need not know whether (G_j^o, C_{j1}^o) was added to S or discarded).

or

(ii) because during the formation of S the pair (G_j^o, C_j^o) was found to exist but discarded.

Assume case (i) holds. Then consider the formation of S'. When we considered a network with C_{j1}^o (before we considered a network with C_j^o), we must have found a network with (G_{j1}^o, C_{j1}^o) such that $G_{j1}^o \leq G_j^o$. Thus no matter whether the network with (G_{j1}^o, C_{j1}^o) was added to S' or discarded, (G_j^o, C_j^o) must be discarded according to the formation rule of S' in Lemma 1 statement (if added, G_j^o in (G_j^o, C_j^o) is not smaller than G_{j1}^o which is in the previously added pair to S'. If discarded, G_j^o in (G_j^o, C_j^o) is still not smaller than the previously added G into S' which is not smaller than G_{j1}^o). Consequently (G_j^o, C_j^o) must not exist in S', contradicting the assumption.

Next assume case (ii) holds. Since (G_j^o, C_j^o) was discarded during the formation of S, S must have contained already another pair (G_{j2}^o, C_{j2}^o) such that $G_{j2}^o < G_j^o$ and $C_{j2}^o \leq C_j^o$. This means that during the formation of S', a pair (G_{j3}^o, C_{j2}^o) such that $G_{j3}^o \leq G_{j2}^o$ must be found. Thus $G_{j3}^o < G_j^o$ and $C_{j2}^o \leq C_j^o$ result. Since these inequalities cannot hold simultaneously by the formation rule of S' in Lemma 1 statement, (G_j^o, C_j^o) must be discarded during the formation of S' and does not exist in S', contradicting the assumption.

Since both cases (i) and (ii) lead to contradiction, some pair in S must contain the C_j^o. Assume that this pair in S has G_{j4}^o which is not equal to G_j^o, i.e., $G_{j4}^o > G_j^o$ or $G_{j4}^o < G_j^o$. But $G_{j4}^o > G_j^o$ cannot occur because during the formation of S the existence of the pair (G_j^o, C_j^o) would prevent (G_{j4}^o, C_j^o) from being contained in S. Also $G_{j4}^o < G_j^o$ cannot occur because during the formation of S', the existence of pair (G_{j4}^o, C_j^o) would prevent (G_j^o, C_j^o) from being contained in S'. Both cases contradict the assumtions.

In conclusion, $G_{j4}^o = G_j^o$ must hold.

This proves that every pair in S' is also in S. The converse statement can be proved similarly. Therefore S' is identical to S.

Q.E.D.

Lemma 2: For any network whose (G, C) is not contained in S, there exists a minimal pair (G_i^o, C_i^o) in S such that $G > G_i^o$, $C \geq C_i^o$ or $G \geq G_i^o$, $C > C_i^o$.

Proof: This is obvious from the way of constructing the minimal pairs in S for f, no matter whether we start with G_g^o (i.e., Definition 2) or with C_c^o (i.e., Lemma 1). Q.E.D.

Theorem 2: Given a monotonically increasing function h(G,C), a network which minimizes the h has some minimal pairs, (G_i^o, C_i^o) in S as only solutions.

Proof: For any pair (G,C) outside S, there is a minimal pair, (G_i^o, C_i^o) such that $G > G_i^o$, $C \geq C_i^o$, or $G \geq G_i^o$, $C > C_i^o$, because of Lemma 2. Thus $h(G,C) > h(G_i^o, C_i^o)$ holds and no pair other than the minimal pairs in S minimizes h. Q.E.D.

Notice that (G_g, C_g) and (G_c, C_c) which are derived for a given switching function for any particular h are not always identical to (G_g^o, C_g^o) and (G_c^o, C_c^o), respectively, because the entire set of minimal pairs in S may not minimize this particular h. If two or more minimal pairs minimize a given h simultaneously, then the property stated in Theorem 1 holds with those minimal pairs.

(G_g^o, C_g^o) and (G_c^o, C_c^o) can be found by finding networks under GCM and CGM, respectively. In the following sections, we will do it in the case of NOR gates, as an example. As we will see, most of the switching functions tested have pairs under GCM which are identical to those under CGM. In other words, for these functions, networks which minimize the number of gates first and the number of connections second also minimize the number of connections first and the number of gates second. Thus for these functions, networks under GCM minimize any monotonically increasing cost function.

Corollary 3: The number of minimal pairs is at most min $(G_c^o - G_g^o + 1,\ C_g^o - C_c^o + 1)$.

Theorem 3: For each minimal pair (G_i^o, C_i^o), there exist an infinite number of monotonically increasing cost functions h(G,C) which are minimized by this pair.

Proof: Consider two planes in the three dimensional space of (h, C, G) which meet at the minimal pair (G_i^o, C_i^o) under consideration:

$$h_1 = C_i^o G + C + G_i^o$$

$$h_2 = G + G_i^o C + C_i^o$$

as shown in Fig. 2.1.

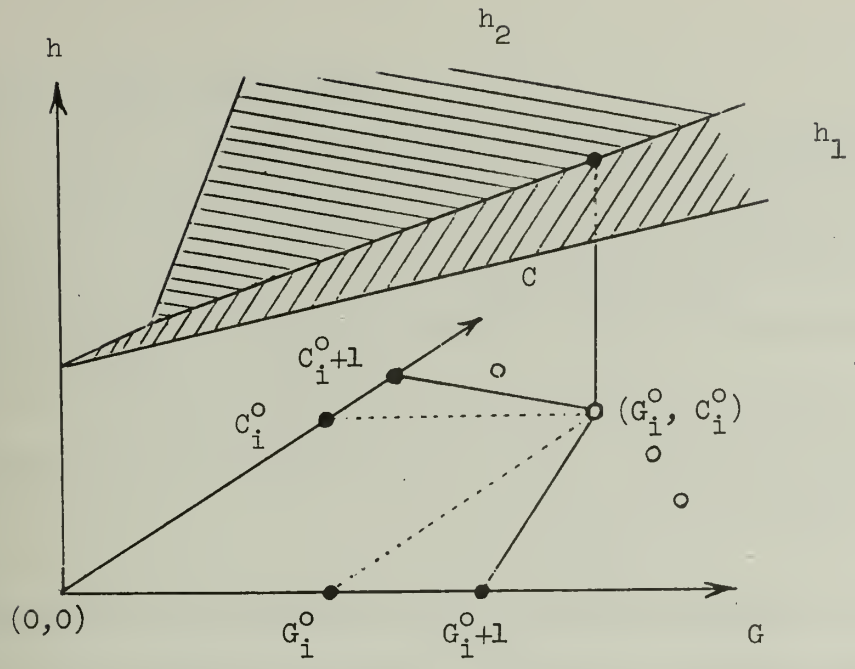


Fig. 2.1. h for Theorem 3.

As can be easily proved, h_1 has the same value on any line in plane h_1 which is parallel to the line from $(G_i^o + 1, 0)$ to (G_i^o, C_i^o). Similarly h_2 has the same value on any line in plane h_2 which is parallel to the line from $(0, C_i^o+1)$ to (G_i^o, C_i^o). Also as can be easily seen from the way of deriving the minimal pairs (shown with small circles in Fig. 2.1), the minimal pairs are outside the quadrangle formed by $(0,0)$, (G_i^o, C_i^o), $(G_i^o + 1, 0)$ and $(0, C_i^o + 1)$. Thus h consisting of h_1 and h_2 (part of h_1 underneath h_2 and part of h_2 underneath h_1 are both ignored) is a monotonically increasing function and has a smaller value for (G_i^o, C_i^o) than those for other minimal pairs.

Even if an arbitrary positive number is multiplied on this h, it is still a monotonically increasing function. Thus there are an infinite number of monotonically increasing functions which are minimized by the pair (G_i^o, C_i^o). Q.E.D.

In passing, let us consider another generalized cost function, i.e., a cost function which is a monotonically non-decreasing function $h'(G,C)$ of G and C, though we will not use it for the rest of this paper. Even a constant number can be such a cost function, i.e., $h'(G,C) = (\text{constant})$, and there are an infinite number of pairs, (G, C)'s which minimize the h'. So generally it is not possible to exhaust all the networks which minimize a monotonically non-decreasing cost function h'.

Theorm 4: Suppose a switching function f and a monotonically non-decreasing cost function $h'(G,C)$ of G and C are given. Then at least some minimal pairs in S minimize the h' (though h' may have other solutions outside S).

Proof: Suppose a network with the pair (G_j, C_j) which minimizes $h'(G, C)$ is found, but (G_j, C_j) is not a minimal pair in S. Because of the way of constructing S, we can always find a minimal pair (G_i^o, C_i^o) in S such that $G_i^o \leq G_j$, $C_i^o < C_j$ or $G_i^o < G_j$, $C_i^o \leq C_j$. Since $h'(G, C)$ is a monotonically non-decreasing function, $h'(G_i^o, C_i^o) \leq h'(G_j, C_j)$ must hold. Consequently, (G_i^o, C_i^o) which is a minimal pair in S must also minimize $h'(G, C)$.

Q.E.D.

Solution By Integer Programming

Turning back to the problem of a monotonically increasing cost function, the problem to obtain the minimal pairs in S can be solved by the logical design procedure based on integer programming.[1]~[10],[16] Here let us discuss how to solve it.

The logical design procedure based on the implicit enumeration method (the implicit enumeration method is a powerful algorithm known for integer programming) yields networks under GCM.[1]~[4], [16] Starting with a sufficiently small number of gates, an integer programming problem whose objective is to minimize the number of connections is formulated. If this problem is feasible, the networks with (G_g^o, C_g^o) are obtained as minimal solutions, otherwise the number of gates in the formulation is increased by one and the integer programming problem is solved. This process is repeated until the problem becomes feasible. This approach is suitable for our goal to obtain the minimal pairs, (G_i^o, C_i^o)'s for a given function f, since we can keep increasing the number of gates in the formulation one by one until networks with (G_c^o, C_c^o) are obtained. All solutions with the number of connections less than the ones obtained by earlier formulations yield the minimal pairs in S. This approach, however, has a disadvantage that (G_c^o, C_c^o) cannot be easily obtained unless a sufficiently large formulation is tested.

The logical design procedure based on the branch-and-bound method [6]~[10], [17] derives minimal networks by a process which is essentially identical to the above approach but uses no inequality formulation and a prespecified gate count. If the cost of a network constructed at some stage of the procedure exceeds the cost of the previously obtained network, the procedure backtracks to search for other networks. By this procedure, all minimal networks under a single given cost function can be obtained at the end of the process. Using $h(G,C) = L \times G + C$ and $h(G,C) = G + L \times C$ as cost functions where L is a sufficiently large positive number, we can obtain minimal networks under GCM and CGM, respectively.

Combining these two methods, we can obtain the set of minimal pairs and the corresponding minimal networks for a given function f as follows.

<u>Procedure to obtain minimal pairs</u>

<u>Step 1</u>: Using the logical design procedure based on the branch-and-bound method, solve minimization problems under GCM and CGM.

<u>Step 2</u>: If $\min (G_c^o - G_g^o + 1, C_g^o - C_c^o + 1) \leq 2$, all the minimal pairs have been obtained by Step 1. Otherwise by the logical design procedure based on the implicit enumeration method, design minimal networks for f for each problem in Definition 2 (i.e., networks with an increasingly greater number of gates which minimize the number of connections) and discard or retain the solutions according to Definition 2. Then we have obtained all the minimal pairs (G_i^o, C_i^o)'s.

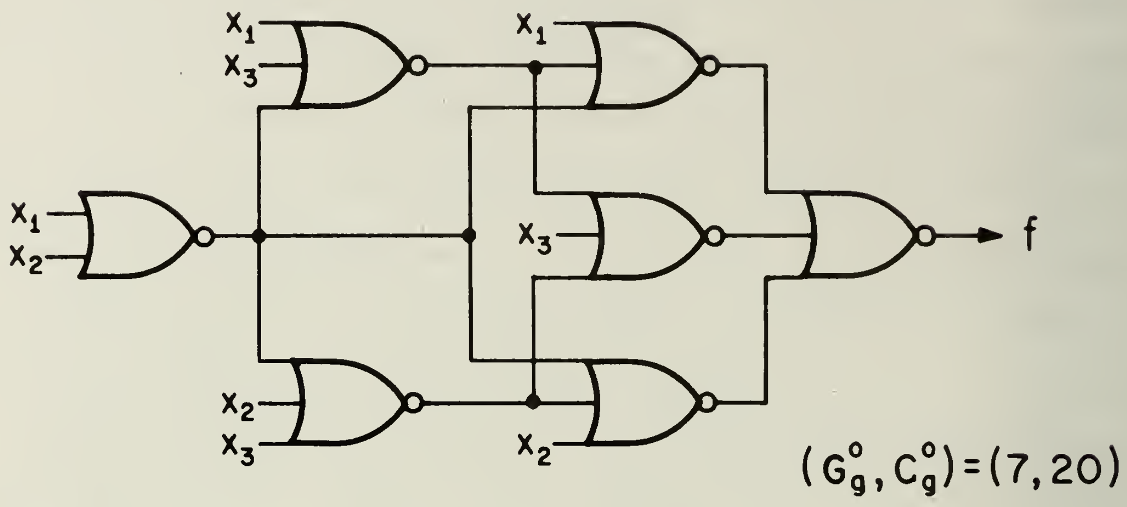
III. MINIMAL NOR NETWORKS FOR ALL FUNCTIONS OF 3 OR LESS VARIABLES

L. Hellerman[12] obtained minimal NOR (also NAND) networks for all functions of 3 or less variables under GCM by an exhaustive method. Although the minimal networks were obtained under the restriction of fan-in and fan-out of three, all networks, except the one for the exclusive-OR function, proved also to be minimal networks under no fan-in and fan-out restrictions by the design based on integer programming.[4] It has been known from [12] and [14] that for the 3-variable exclusive-OR function, the minimal NOR network under GCM which consists of 7 gates and 20 connections as shown in Fig. 3.1 (a) is not the only minimal network under the monotonically increasing cost function $h(G,C) = 4G + C$, for example, since a network consisting of 8 NOR-gates and 16 connections can be obtained by cascading two 2-variable exclusive-NOR networks as shown in Fig. 3.1 (b). Although neither of the two papers [12] and [14] mentioned the minimality of the latter network, it is a convincing example for the fact that the minimal networks under different minimization criteria can be different from each other. However, the computational results show that among 77 non-trivial (excluding constants and input variable itself) P-equivalence class representative functions* of 3 or less variables, there are only two functions for which the minimal networks under GCM and CGM are different from each other. The two representative functions are as follows:

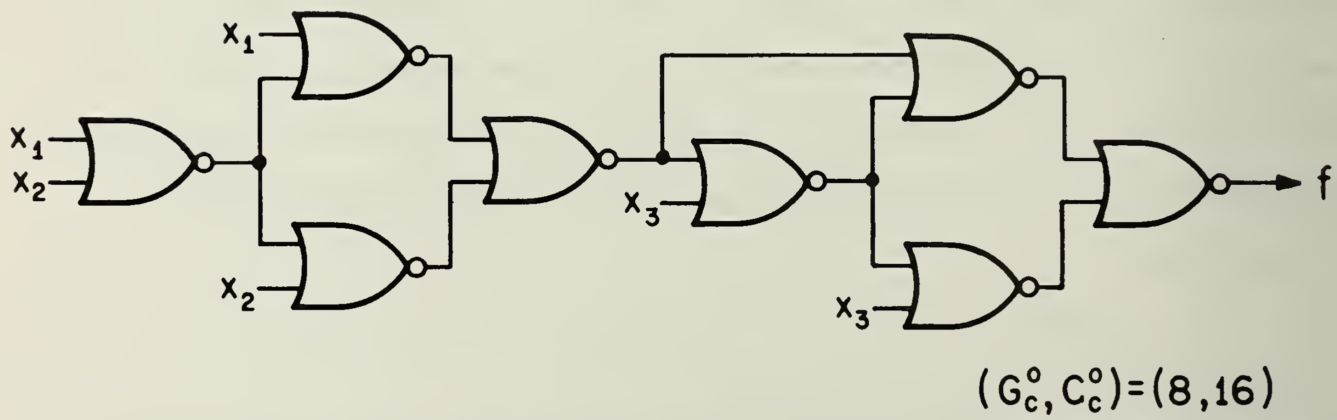
(1) $f = x_1 \oplus x_2 \oplus x_3$

*Two functions are defined to be in the same P-equivalence class if and only if one function can be obtained from the other by permutation of input variables. (See [2], for example.)

Fig. 3.1 Minimal networks corresponding to the minimal pairs (G_i^o, C_i^o) for function $f = x_1 \oplus x_2 \oplus x_3$

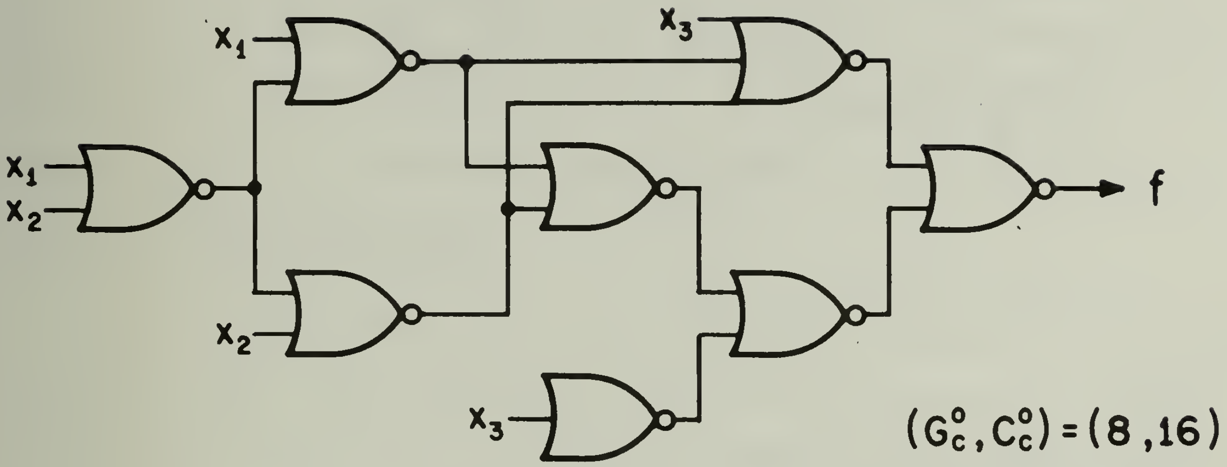


(a) minimal network under GCM.

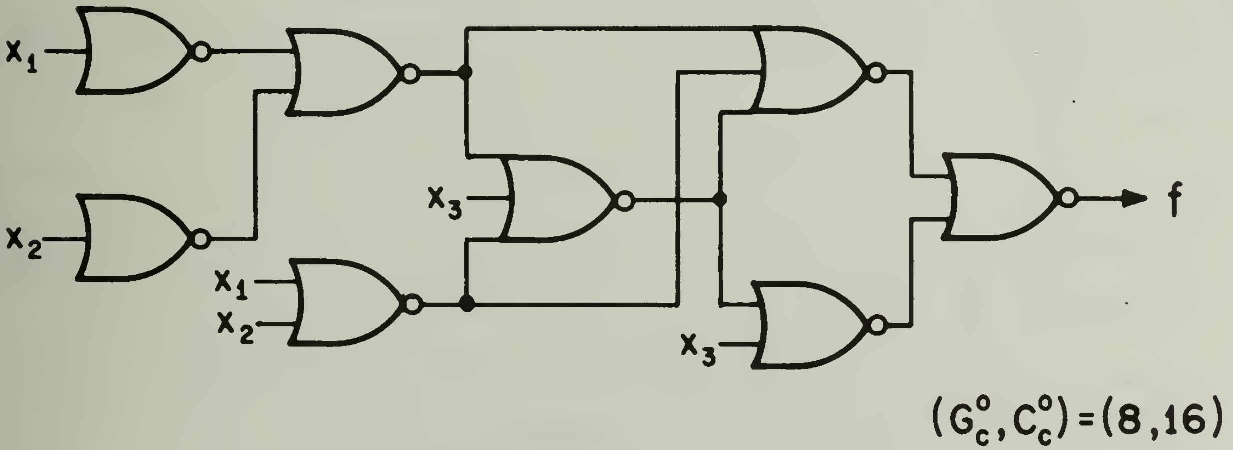


(b) minimal network under CGM.

ig. 3.1 (continued)

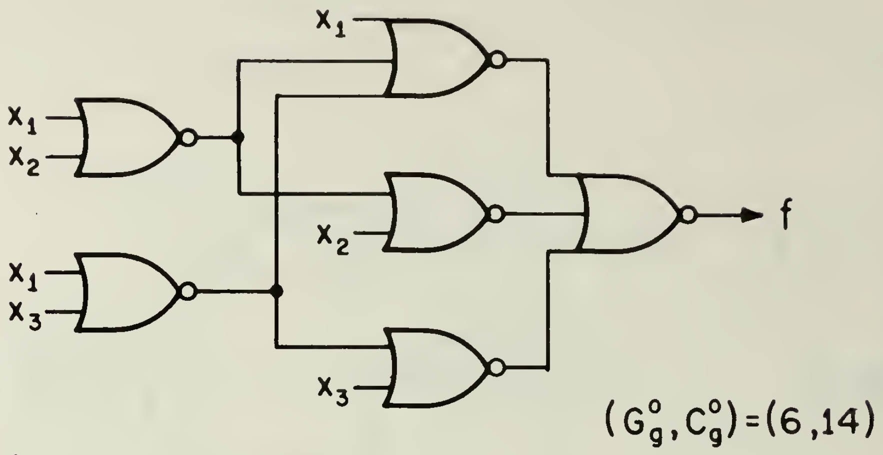


(c) minimal network under CGM.

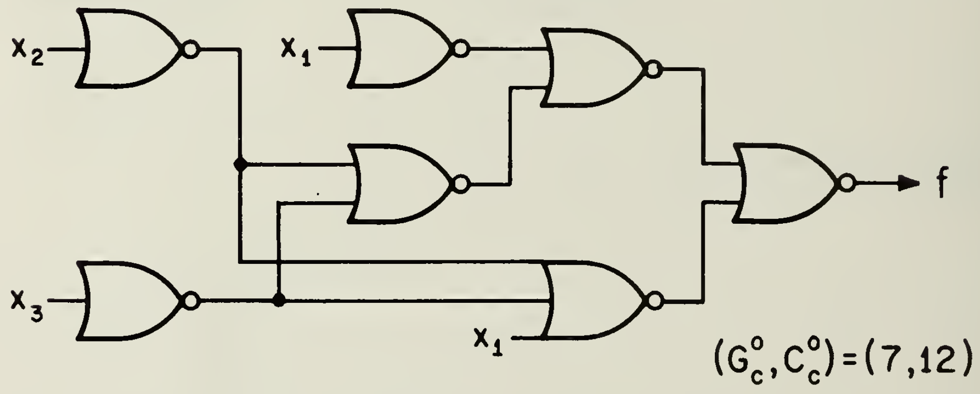


(d) minimal network under CGM.

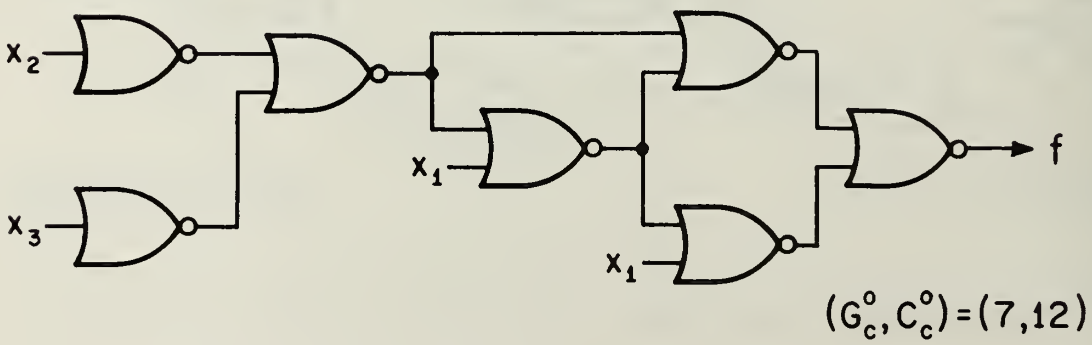
Fig. 3.2 Minimal networks for function $f = x_1x_2x_3 \vee \bar{x}_1(\bar{x}_2 \vee \bar{x}_3)$



(a) minimal network under GCM.



(b) minimal network under CGM.



(c) minimal network under CGM.

The minimal NOR network under GCM contains 7 gates and 20 connections as shown in Fig. 3.1.(a), whereas the minimal NOR networks under CGM contain 8 gates and 16 connections as shown in Fig. 3.1. (b), (c) and (d). Since min $\{G_c^o - G_g^o + 1, C_g^o - C_c^o + 1\} = 2$, by Corollary 3, S contains only two minimal pairs, namely, $(G_g^o, C_g^o) = (7,20)$ and $(G_c^o, C_c^o) = (8,16)$. The set of minimal networks therefore, contains only 4 networks shown in Fig. 3.1 (a) through (d).

(2) $f = x_1x_2x_3 \vee \bar{x}_1(\bar{x}_2 \vee \bar{x}_3)$

The minimal NOR network under GCM contains 6 gates and 14 connections as shown in Fig 3.2 (a), whereas the minimal NOR network under CGM, contains 7 gates and 12 connections as shown in Fig. 3.2(b) and (c). From Corollary 3, $\min\{G_c^o - G_g^o + 1, C_g^o - C_c^o + 1\} = \min\{2,3\} = 2$, and therefore the set of minimal networks for this function contains only the above three networks with pairs $(G_g^o, C_g^o) = (6, 14)$ and $(G_c^o, C_c^o) = (7, 12)$.

The computation was done on IBM 360/75J by the program ILLOD-(NOR-B) [8], [9], [10] assuming only uncomplemented variables being available as inputs to a network and no fan-in, fan-out or level restrictions. Program ILLOD-(NOR-B) is a FORTRAN program for designing minimal NOR networks under linear cost functions based on the branch-and-bound algorithm [8], [10]. In general, the program during its execution must enumerate much more network configurations when it attempts to obtain minimal networks under CGM than it does under GCM, and therfore the computation time under CGM is usually much longer than that under GCM. Table 3.1 shows the statistics of the computation times and the numbers of backtracks for 13 P-equivalence representative functions of three variables which require at least 12 connections in their minimal networks under CGM. As Table 3.1 shows, the computation time is excessively long for minimal networks under CGM, especially for functions requiring a large number of connections. An interesting observation is that the computation time under CGM depends primarily on the number of connections in a minimal network under CGM whereas the computation time under GCM depends primarily on the number of gates in a minimal network under GCM.

Function	Connection-gate minimization (CGM)			Gate-connection minimization (GCM)		
	minimal network (G_c^o, C_c^o)	computation time in sec.*	number of backtracks	minimal network (G_g^o, C_g^o)	computation time in sec.*	number of backtracks
OCT 201	(5,12)	14.20	1374	(5,12)	0.33	29
OCT 152	(6,12)	3.23	277	(6,12)	1.37	71
OCT 207	(7,12)	10.33	836	(6,14)	0.80	37
OCT 026	(6,14)	159.43	13991	(6,14)	2.20	182
OCT 051	(7,14)	44.53	3906	(7,14)	3.44	313
OCT 351	(5,15)	207.79	16274	(5,15)	0.22	9
OCT 206	(6,15)	223.75	17592	(6,15)	3.56	267
OCT 236	(6,15)	129.80	9991	(6,15)	0.82	24
OCT 150	(7,15)	57.59	5530	(7,15)	3.33	201
OCT 153	(7,15)	94.43	8306	(7,15)	2.29	142
OCT 227	(7,15)	89.25	6685	(7,15)	1.85	113
OCT 151	(7,16)	198.78	15778	(7,16)	2.19	117
OCT 226	(8,16)	442.40	32069	(7,20)	2.26	147
TOTAL		1675.51	132609		24.66	1654
AVERAGE		128.89	10200.7		1.90	127.2

Table 3.1 Detailed statistics for 13 functions of 3 variables which require 12 or more connections under CGM.
Function is expressed with the function column of the truth table in octal form, e.g., OCT 152 = $(\underbrace{010}_{2} \ \underbrace{101}_{5} \ \underbrace{10}_{1})$ which corresponds to

$f = \Sigma(1,3,5,6)$

* IBM 360/75J with FORTRAN Compiler H level 21.7 (opt 2), using the University of Illinois system-provided timing subroutine with entry names STIMEZ and KTIMEZ.

IV. MINIMAL NOR NETWORKS FOR SOME 4-VARIABLE FUNCTIONS

There are 3984 P-equivalence classes of switching functions of four or less variables. 3904 of them are P-equivalence classes of functions of exactly four variables. For these four variable functions, Ikeno, Hashimoto and Naito obtained minimal NAND networks under h(G,C) = G for all four variable functions requiring at most seven NAND gates[15] (the number of connections is not necessarily minimized in each network). Later, J. Culliney, using an integer programming approach, solved all four variable functions which require at most five NOR-gates under GCM [16]. The second and the third columns in Table 4.1 are taken from [16], that show the number of P-equivalent classes which require R NOR gates under GCM for R = 1, 2, ..., 7, and ≥ 7, and the average number of connections for each group, respectively.

Like the observation in the previous section, the amount of computation time required for determining minimal networks under CGM for a given function is mainly determined by the number of connections in the corresponding minimal networks. Since the minimal number of connections is not known before the function is actually solved under CGM, it is difficult to estimate the required computation time beforehand. However, the number of connections in a minimal network under GCM gives a very good upper bound. Since computation time needed for determining minimal networks increases rapidly with the increase of the sizes of minimal networks (i.e., the number of gates and the number of connections), only the functions requiring five or less gates in their minimal gate networks under GCM were solved under CGM. Table 4.1 shows some of the results. For functions requiring less than four gates, the minimal networks under GCM are identical to those under CGM. Out of 60 P-equivalence class representative functions requiring 4 NOR gates, 2 functions have minimal networks under CGM different from minimal networks under GCM.

number of gates in minimal network under GCM	number of P-equivalence classes	average number of connections in minimal networks under GCM	number of P-equivalence classes which require less connection under CGM than under GCM	average number of connections in minimal networks under CGM
1	1	4.0	0	4.0
2	4	5.0	0	5.0
3	13	6.3	0	6.3
4	60	9.0	2	8.9
5	234	11.5	24	11.3
6	707	14.3*		
7	1334	16.5**		
≥ 7	1551			

*Average number is obtained by solving 83 sample functions.

**Average number is obtained by solving 46 sample functions.

Table 4.1 Satistics on the minimal networks under GCM and CGM for 4-variable functions.

However, for both functions, min $\{G_c^o - G_g^o + 1, C_g^o - C_c^o + 1\} = 2$ holds so that the minimal networks under GCM and CGM are the only minimal networks corresponding to the minimal pairs (G_i^o, C_i^o) for these two functions. Similarly, out of 234 P-equivalence class representative functions requiring 5 NOR gates, 24 functions* require less connections than their minimal networks under GCM. For each of 21 such functions, min $\{G_c^o - G_g^o + 1, C_g^o - C_c^o +1\} = 2$ holds, so the numbe of the minimal pairs is exactly two. For the other three functions, however, min $\{G_c^o - G_g^o + 1, C_g^o - C_c^o + 1\} = 3$, and therefore more computation was necessary to determine all the minimal pairs, (G_i^o, C_i^o). The three functions and the corresponding minimal pairs are as follows:

(1) $f = \Sigma(0,7,8,9,10,11,12,13,14,15)$, $(G_g^o, C_g^o) = (5,15)$,

$(G_c^o, C_c^o) = (7,13)$,

(2) $f = \Sigma(0,4,7,8,9,10,11,12,13,14,15)$, $(G_g^o, C_g^o) = (5,14)$,

$(G_c^o, C_c^o) = (7,12)$,

(3) $f = \Sigma(0,3,5,6,7,8,9,10,11,12,13,14,15)$, $(G_g^o, C_g^o) = (5,18)$,

$(G_c^o, C_c^o) = (7,16)$.

Since min $\{G_c^o - G_g^o + 1, C_g^o - C_c^o + 1\} > 2$ holds in each of the above three cases, the integer programming approach should be used to find other minimal

*Because of excessively long computation time under CGM for one of the 24 functions $f = \Sigma(0,5,6,7,9,10,11,12,13,14,15)$ whose $(G_g^o,C_g^o) = (5,18)$, the minimality of pair (7,17) under CGM has not been proved. However, it has been proved that $C_c^o \geq 16$ and that there is no solution with pairs (6,16), (7,16), (8,16) or (9,16). Thus (7,17) is very likely the minimal pair (G_c^o,C_c^o) for this function and the later discussion assumes $(G_c^o,C_c^o) = (7,17)$ for this function.

pairs. The only possible minimal pairs are (6,14), (6,13), and (6,17) in cases (1), (2), and (3), respectively. But these pairs were found not to exist (ILLOD(NOR-B) was used to solve these problems with the minimization of $h(G,C) = G + C$, instead of using the design procedure based on the implicit enumeration method as mentioned in Step 2 in the procedure at the end of Section II. This was because $G_g^o + C_g^o = G_c^o + C_c^o = G_p^o + C_p^o$ holds in each of the above three cases, where (G_p^o, C_p^o) is (6,14), (6,13), or (6,17).) From these results it is concluded that for all 4-variable functions requiring 5 or less gates,the set of S contains at most two (G,C)'s (i.e., (G_g^o, C_g^o) and (G_c^o, C_c^o)).

Some statistics on the computation time in seconds by program ILLOD(NOR-B) on IBM 360/75J are shown in Table 4.2. It is again concluded from this table that the computation times under CGM are primarily determined by the number of connections in the minimal networks under CGM, whereas the computation times under GCM depend on the number of gates in the minimal networks under GCM. Another observation is that the computation time increases very rapidly as the number of connections increases. In fact, the computation time increases more rapidly than exponentially as is shown by the last column in Table 4.2, which is computation time increasing ratio obtained by dividing the average computation time for a given number of connections by the computation time for one less number of connections. Table 4.2 does not include the computation time for the function $f = \Sigma\,(0,5,6,7,9,10,11,12,13,14,15)$ (See the footnote earlier in this section.)

The functions which require less connections than their minimal network under GCM are listed in Appendix along with their minimal networks under the two different minimization criteria.

The percentage of the functions which have different minimal networks under GCM and CGM appears to increase as functions require more gates in their minimal networks under GCM. The results shown in Table 4.1 appear to indicate this tendency. Among 60 functions requiring 4 gates, only two functions have minimal networks under CGM which are different from minimal networks under GCM, i.e., the ratio of these functions to others is 1/30. In the case of functions requiring 5 gates, the ratio increases to more than 1/10, i.e., 24 out of 234 functions belong to this catagory.

number of connections \ number of gates	1	2	3	4	5	6	7	Average	Ratio of computation times
4	0.13 (1)							0.13 (1)	
5		0.14 (3)						0.14 (3)	1.1
6			0.19 (3)					0.19 (3)	1.3
7			0.28 (2)	0.31 (17)				0.30 (19)	1.6
8			0.37 (1)	0.57 (10)	0.61 (18)			0.58 (29)	1.9
9				1.2 (9)	1.3 (21)			1.3 (30)	2.2
10				3.1 (11)	2.7 (34)	3.6 (3)		2.8 (48)	2.2
11				9.0 (8)	7.3 (47)	9.0 (4)		7.6 (59)	2.7
12				26 (3)	22 (45)	39 (1)	35 (5)	23 (54)	3.1
13					78 (28)		14×10 (4)	86 (32)	3.7
14					35×10 (11)	25×10 (1)	40×10 (2)	35×10 (14)	4.1
15					88×10 (3)		12×10^2 (2)	99×10 (5)	2.9
16					35×10^2 (3)		33×10^2 (3)	34×10^2 (6)	3.4

Table 4.2. Average Computation Time* in seconds on IBM 360/75J under CGM for Four-Variable Functions Requiring at most 5 NOR Gates (The numbers in parentheses show the numbers of P-equivalence classes.)

* See the note in Table 3.1

V. CONCLUSION

After discussing that the minimization of a monotonically increasing cost function h(G,C) of gate count G and connection count C would ususally lead to a compact network on an IC chip, it was shown that all the minimal solutions of the h(G,C) must be found in the minimal pairs, (G_i^o, C_i^o)'s, which can be derived by solving the logical design problems based on integer programming. Then in the case of NOR combinational networks, as an example, the minimal pairs were calculated for all functions of three or less variables and also some functions of four variables which under GCM require 5 or less NOR gates. The computational results have indicated that for most of these functions, NOR networks with the number of gates minimized as the primary objective and with the number of connections as the secondary objective, have also minimized the number of connections as the primary objective and the number of gates as the secondary objective. So if we want to have a simple design criterion for compact networks (even though many complex IC implementation restrictions must be yet considered during logical design), conventional design criterion, i.e., the minimization of the number of gates as the primary objective and the minimization of the number of connections as the secondary objective would be reasonable at least in the case of small NOR networks. (These small networks are the practical case where the minimization is feasible since the minimization is not possible in the case of large networks anyway. In practice, large networks, if we want to minimize, must be decomposed into small networks of manageable size.) This is the useful conclusion since conventional design criterion, i.e., GCM is much easier to use and less time-consuming than CGM. Although there may be some exceptions to this general tendency, such exceptions would not occur so often and even if they occur, the deviation of the chip area of a designed network would not be too far. Of course a solid conclusion could be obtained only after the accumulation of empirical observations by actual layouts.

When many connections run over gate areas, the above conclusion is further strengthened. Also the fact that the elimination of a gate eliminates the power supply and ground connections and also a load MOSFET strengthens the above conclusion (this is true particularly in the case of MOS. A load MOSFET occupies a larger area than a driver MOSFET in the case of static MOS. Clock supply connections are eliminated in the case of dynamic MOS).

The choice of NOR networks as a computational example is meaningful because NOR gates are still basic logic gates in many cases and especially because NOR networks are a basis of logical design with ECL gate packages and are closely related to integrated injection logic networks (as discussed in Section II). However for the cases when networks consist of gates other than NOR gates (particularly when we use MOS gates which can express more complex functions than NOR), we would need further experiments in the future.

VI. ACKNOWLEDGEMENT

The authors are grateful to T. Nakagawa for his help in the initial setup for the computation and also Y. Muroga for her error checking of the table in Appendix.

REFERENCES

1. S. Muroga, "Logical Design of Optimal Digital Networks by Integer Programming", Advances in Information System Science, Vol. 3, edited by J.T. Tou, Plenum Press, pp. 283-348, 1970.

2. S. Muroga, "Threshold Logic and its Applications", Wiley-International, chapter 14, 1971.

3. S. Muroga and I. Ibaraki, "Design of Optimal Switching Networks by Integer Programming", IEEE Vol. C-21, No. 6, pp. 573-582, June, 1972.

4. C.R. Baugh, I. Ibaraki, T.K. Liu, and S. Muroga, "Optimum Network Design Using NOR and NOR-AND Gates by Integer Programming", Report No. 293, Department of Computer Science, Univ. of Illinois, January, 1969.

5. T.K. Liu, K. Hohulin, L.E. Shiau, and S. Muroga, "Logical Design of Optimal One-Bit Full Adder by Integer Programming", IEEE Vol. C-23, No. 1, pp. 63-70, Jan., 1974.

6. E.S. Davidson, "An Algorithm for NAND Decomposition of Combinational Switching Functions", Ph.D. Thesis, Department of Electrical Engineering and Coordinated Science Laboratory, University of Illinois, 1968.

7. E.S. Davidson,"An Algorithm for NAND Decomposition under Network Constraints", IEEE, Vol. C-18, No. 12, pp. 1098-1109, December, 1969.

8. T. Nakagawa and H.C. Lai, "A Branch-AND-Bound Algorithm for Optimal NOR Networks (The Algorithm Description)", Report No. 438, Department of Computer Science, University of Illinois, April, 1971.

9. T. Nakagawa, H.C. Lai, and S. Muroga, "Pruning and Branching Methods for Designing Optimal Networks by the Branch-And-Bound Method", Report No. 471, Department of Computer Science, University of Illinois, August, 1971.

10. T. Nakagawa, and H.C. Lai, "Reference Manual of FORTRAN Program ILLOD(NOR-B) for Optimal NOR Networks", Report No. 488, Department of Computer Science, University of Illinois, December, 1971.

11. Quine-McClusky Method. See for example, Introduction to the Theory of Switching Circuits, by E.J. McClusky, McGraw Hill Co., 1965.

12. L. Hellerman, "A Catalog of Three-Variable Or-Invert and And-Invert Logical Circuits", IEEE Vol. EC-12, No. 3, pp. 198-223, June, 1963.

13. J. Gimple, "The Minimization of TANT Networks", IEEE Vol. EC-16, No. 1, pp. 18-39, February, 1967.

14. G. Maley, "Manual of Logic Circuits", Prentice-Hall, 1970.

15. N. Ikeno, A. Hashimoto and K. Naito, "A Table of Four-Variable Minimal NAND Circuits", Electrical Communication Laboratory Technical Journal, Extra issue No. 26, Nippon Telegraph and Telepone Public Corporation, Tokyo, Japan, 1968 (in Japanese).

16. J. Culliney, "On the Synthesis by Integer Programming of Optimal NOR Gate Networks for Four-Variable Switching Functions", Report No. 480, Department of Computer Science, University of Illinois, 1971.

17. H.C. Lai, T. Nakagawa, and S. Muroga, "Redundancy Check Technique in Designing Optimal Networks by Branch-And-Bound Method", to be published in International Journal of Computer and Information Sciences, 1974.

18. H. Hart and A. Slob, "Integrated Injection Logic: A New Approach to LSI", IEEE, Journal of Solid State Circuits, pp. 346-351, October, 1972.

19. H.H. Berger and S.K. Wiedman, "Merged-Transistor Logic--A Low Cost Bipolar Logic Concept", IEEE, Journal of Solid State Circuits, pp. 340-346, October, 1972.

20. American Micro-Systems, Inc., "MOS Integrated Circuits", Van Nostrand, 1972.

21. Electronics, pp. 91-96, February 21, 1974.

APPENDIX MINIMAL NOR NETWORKS UNDER GCM AND CGM FOR 4-VARIABLE FUNCTION WHICH UNDER GCM REQUIRE 5 OR LESS GATES

Among 312 representative functions of P-equivalence classes of 4-variable requiring at most 5 NOR gates under GCM, 26 functions have minimal networks under CGM which are different from their corresponding minimal networks under GCM. The following is the list of these functions and their minimum networks.

Description of the catalog

To the left of the vertical line, minimal NOR networks under GCM for each function f are shown, and to the right those under CGM for the same f are shown. A function f is represented by the value of f in its truth table expressed in the hexadecimal form, where

0000	0
0001	1
0010	2
.	.
.	.
.	.
.	.
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

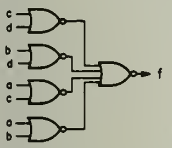
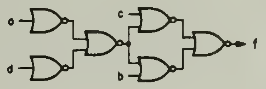
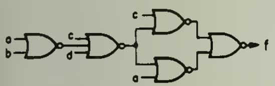
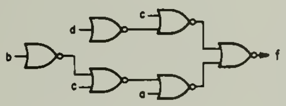
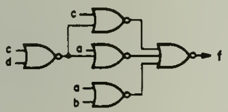
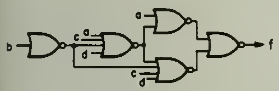
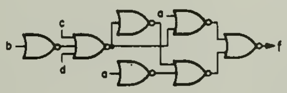
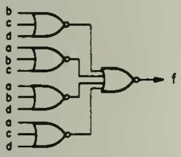
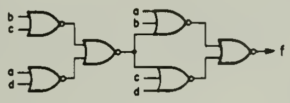
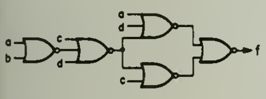
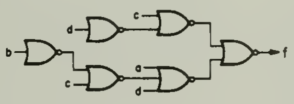
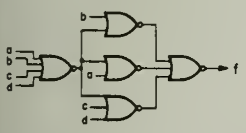
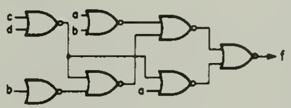
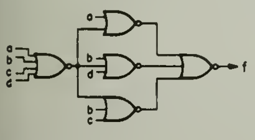
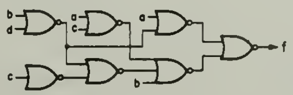
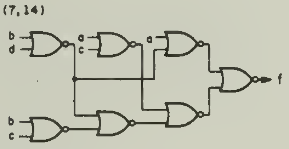
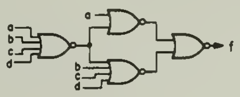
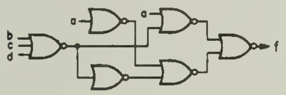
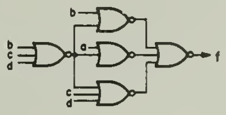
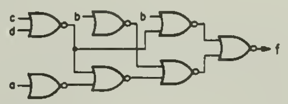
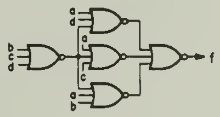
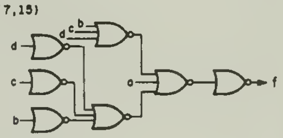
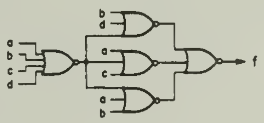
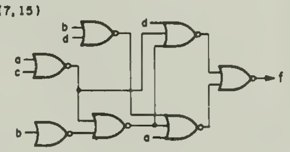
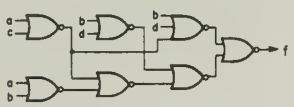
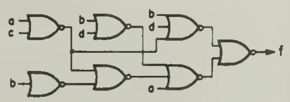
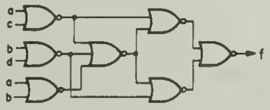
For example f = (0357) represents the following

$x_1x_2x_3x_4$	f	
0 0 0 0	0	
0 0 0 1	0	0
0 0 1 0	0	
0 0 1 1	0	
0 1 0 0	0	
0 1 0 1	0	3
0 1 1 0	1	
0 1 1 1	1	
1 0 0 0	0	
1 0 0 1	1	5
1 0 1 0	0	
1 0 1 1	1	
1 1 0 0	0	
1 1 0 1	1	7
1 1 1 0	1	
1 1 1 1	1	

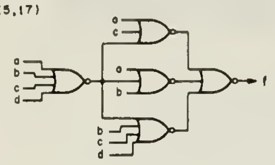
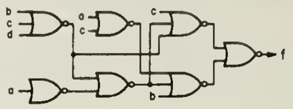
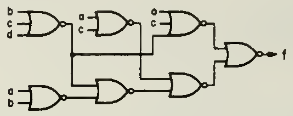
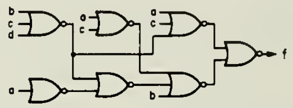
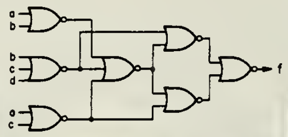
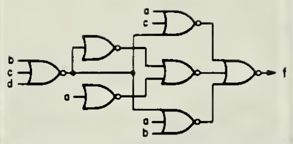
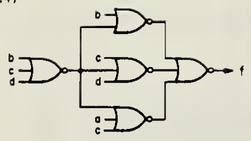
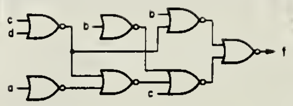
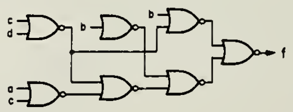
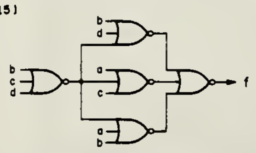
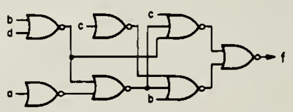
a, b, c, d are used instead of x_1, x_2, x_3, x_4 for the sake of photo reproduction.

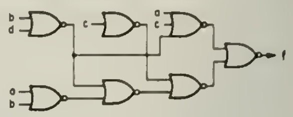
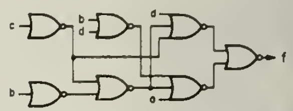
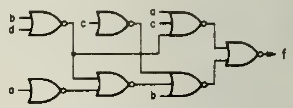
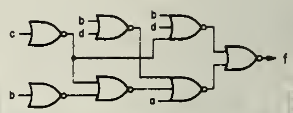
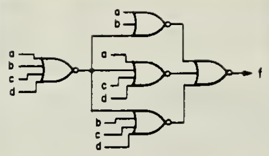
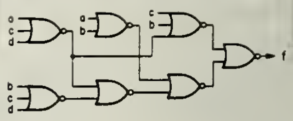
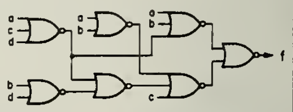
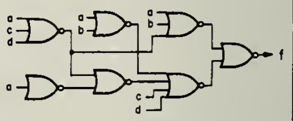
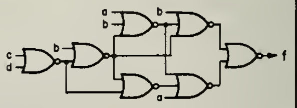
Pairs (G_g^o, C_g^o)'s are shown with minimal networks under GCM, and pairs (G_c^o, C_c^o)'s are shown with those under CGM.

TABLE OF MINIMAL NOR NETWORKS UNDER GCM AND CGM FOR 4-VARIABLE FUNCTIONS REQUIRING 5 OR LESS GATES

MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
1. f = (0357)	
(5,12) 	(6,10) 
2. f = (08B8)	
(5,11) 	(6,10) 
(5,11) 	
3. f = (08F7)	
(5,15) 	(7,12) 
4. f = (177F)	
(5,16) 	(6,14) 
5. f = (1988)	
(5,12) 	(6,11) 
6. f = (8007)	
(5,14) 	(7,15) 
7. f = (801F)	
(5,15) 	(7,14) 
	(7,14) 
8. f = (B07F)	
(4,12) 	(6,11) 
9. f = (8087)	
(5,15) 	(7,12) 
10. f = (81FF)	
(5,15) 	(7,15) 
11. f = (B35F)	
(5,16) 	(7,15) 
	(7,15) 
	(7,15) 
	(7,15) 

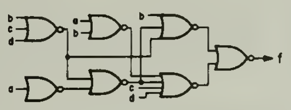
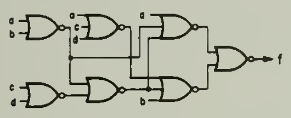
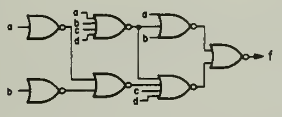
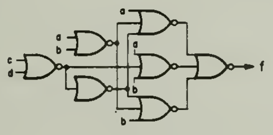
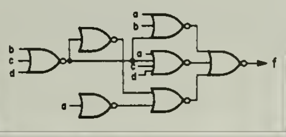
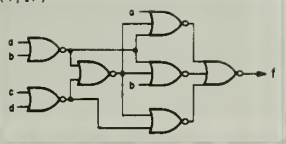
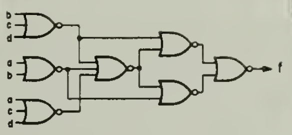
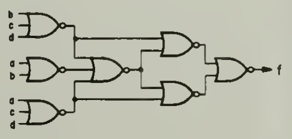
(CONTINUED)

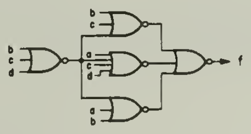
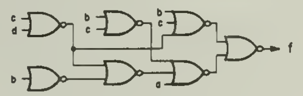
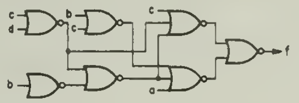
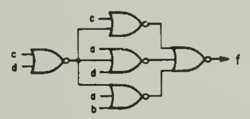
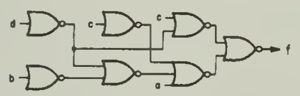
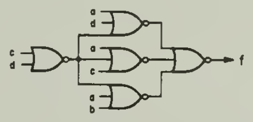
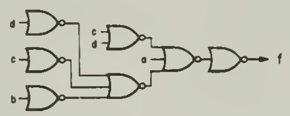
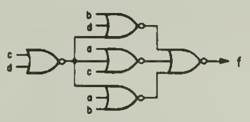
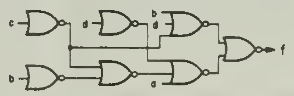
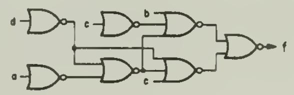
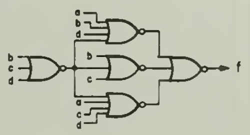
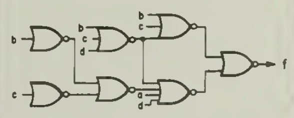
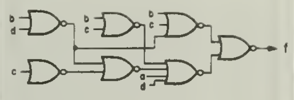
MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
12. f=(837F)	
(5,17) 	(7,16) 
	(7,16) 
	(7,16) 
	(7,16) 
	(7,16) 
13. f=(8387)	
(5,14) 	(7,13) 
	(7,13) 
14. f=(83DF)	
(5,15) 	(7,14) 

MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
14. f=(83DF) (CONTINUED)	
	(7,14) 
	(7,14) 
	(7,14) 
	(7,14) 
15. f=(877F)*	
(5,18) 	(7,17) 
	(7,17) 
	(7,17) 
	(7,17) 

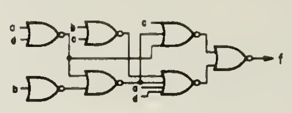
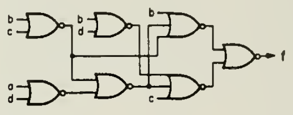
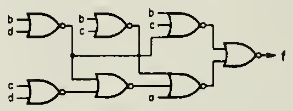
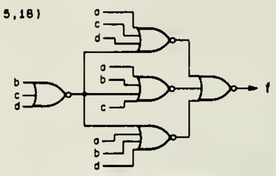
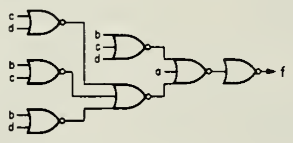
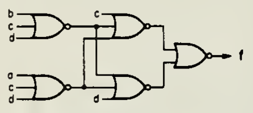
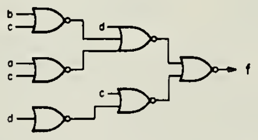
* (7,17) IS BELIEVED TO BE (G_c^0,C_c^0). SEE THE FOOTNOTE IN SECTION IV.

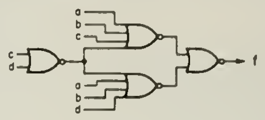
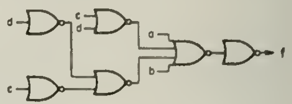
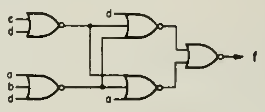
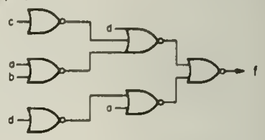
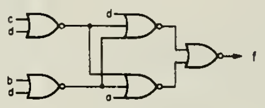
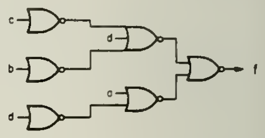
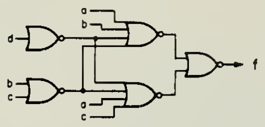
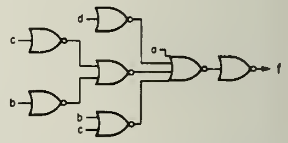
(CONTINUED)

MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
15. f = (877F) (CONTINUED)	
	(7,17) 
	(7,17) 
	(7,17) 
	(7,17) 
	(7,17) 
	(7,17) 
	(7,17) 
	(7,17) 

MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
16. f = (87BF)	
(5,16) 	(7,15) 
	(7,15) 
17. f = (89BB)	
(5,13) 	(7,12) 
18. f = (89FF)	
(5,14) 	(7,12) 
19. f = (88DF)	
(5,14) 	(7,13) 
	(7,13) 
20. f = (97BF)	
(5,17) 	(7,16) 
	(7,16) 

(CONTINUED)

MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
20. f = (978F) (CONTINUED)	
	(7,16) 
	(7,16) 
	(7,16) 
21. f = (97FF)	
(5,18) 	(7,16) 
22. f = (9991)	
(5,14) 	(6,12) 

MINIMAL NOR NETWORKS UNDER GCM	MINIMAL NOR NETWORKS UNDER CGM
23. f = (9FFF)	
(4,12) 	(6,11) 
24. f = (A8DD)	
(5,13) 	(6,11) 
25. f = (A8FD)	
(5,12) 	(6,10) 
26. f = (EBFF)	
(5,13) 	(7,12) 

IBLIOGRAPHIC DATA HEET	1. Report No. UIUCDCS-R-74-649	2.	3. Recipient's Accession No.
Title and Subtitle MINIMIZATION OF LOGIC NETWORKS UNDER A GENERALIZED COST FUNCTION			5. Report Date April, 1974
			6.
Author(s) Saburo Muroga and Hung Chi Lai			8. Performing Organization Rept. No.
Performing Organization Name and Address Department of Computer Science University of Illinois at Champaign-Urbana Urbana, Illinois 61801			10. Project/Task/Work Unit No.
			11. Contract/Grant No. NSF Grant No. NSF-GJ-40221
Sponsoring Organization Name and Address National Science Foundation 1800 G. Street, N.W. Washington, D.C. 20550			13. Type of Report & Period Covered Technical Report
			14.
Supplementary Notes			

Abstracts Based on the intuitive observation that smaller numbers of gates and connecions would usually lead to a more compact network on an integrated circuit, a monotoncally increasing function of gate count and connection count is concluded to be a easonable cost function to be minimized in logical design of a network implemented in C. Then it is shown that all minimal solutions of such a cost function can be always ound among the following: minimal networks with a minimal number of gates as the irst objective and a minimal number of connections as the second objective; minimal etworks with a minimal number of connections as the first objective and a minimal umber of gates as the second objective; and minimal networks which are associated to ne above two types of minimal networks. All these three types of minimal networks of OR gates, as an example, are calculated by the logical design programs based on nteger programming, for all functions of three or less variables and also some funcions of four variables which require 5 or less NOR gates. According to the computational results, for the majority of the functions the first type minimal networks re identical to the second type, and for no function were networks of the third type ound to exist.

7. Key Words and Document Analysis. 17a. Descriptors

ogical design, Integrated circuit, Generalized cost function, Minimal network, ECL, TL, MOS, IIL, Integer programming, Implicit enumeration method, Branch-and-bound ethod, NOR gates, Minimization of the number of gates, Minimization of the number f connections.

7b Identifiers/Open-Ended Terms

7c COSATI Field/Group

vailability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 40
	20. Security Class (This Page UNCLASSIFIED	22. Price